# NAVAL POSTGRADUATE SCHOOL
# MONTEREY, CALIFORNIA


# THESIS

**OPTIMIZING ASW SEARCH FOR HVU PROTECTION USING THE FAB ALGORITHM**

by

Masaaki Kunigami

March 1997

Thesis Advisor: Alan R. Washburn

**Approved for public release; distribution is unlimited**

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY CA 93943-5101

# REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188



| 1. AGENCY USE ONLY *(Leave blank)* | 2. REPORT DATE March 1997 | 3. REPORT TYPE AND DATES COVERED Master's Thesis |
|---|---|---|

4. TITLE AND SUBTITLE OPTIMIZING ASW SEARCH FOR HVU PROTECTION USING THE FAB ALGORITHM

5. FUNDING NUMBERS

6. AUTHOR(S) KUNIGAMI, Masaaki

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

This thesis researches the feasibility of a TDA (tactical decision aid) to defend a high value surface unit from an enemy submarine. Accordingly, this research adopts a FAB (forward and backward) algorithm to search for a moving target. It develops a prototype of a TDA: FABTDA which gives an optimal allocation for search aircraft.

14. SUBJECT TERMS
Optimal search; FAB algorithm; dynamic programming;

15. NUMBER OF PAGES 40

16. PRICE CODE

| 17. SECURITY CLASSIFICATION OF REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
|---|---|---|---|
| Unclassified | Unclassified | Unclassified | UL |

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18 298-102

Approved for public release; distribution is unlimited.

# OPTIMIZING ASW SEARCH FOR HVU PROTECTION USING THE FAB ALGORITHM

Masaaki Kunigami
Technical Official 2nd Grade, Defense Agency Japan
M.E.,Kyushu University, 1990

Submitted in partial fulfillment
of the requirements for the degree of

**MASTER OF SCIENCE IN OPERATIONS RESEARCH**

from the

**NAVAL POSTGRADUATE SCHOOL**
**March 1997**

## ABSTRACT

This thesis researches the feasibility of a TDA (tactical decision aid) to defend a high value surface unit from an enemy submarine. Accordingly, this research adopts a FAB (forward and backward) algorithm to search for a moving target. It develops a prototype of a TDA: FABTDA which gives an optimal allocation for search aircraft.

## TABLE OF CONTENTS

## ACKNOWLEDGMENTS

This thesis owes much to the thoughtful and helpful advice of Prof. Alan Washburn. Hereby, the author wants to thank him for his guidance and patience. Thanks are also due to LCDR John Hosford for reading the draft and making valuable suggestions. The author wants to thank Prof. Siriphong Lawphongpanich for his valuable advice and CDR Robert Handlers for reading the entire text and making helpful suggestions.

# I. INTRODUCTION

## A. BACKGROUND

When broad localization of a hostile submarine is given, a fleet commander who commands a high value surface unit (HVU) faces the problem of whether he should use his airborne ASW (Anti Submarine Warfare) units in an offensive or defensive posture against the target. If the initial information is accurate enough, concentration of ASW assets for killing the target could be effective for both defending the HVU as well as prosecuting the target. Therefore, it would be logical if the development of a TDA (tactical decision aid) included a defensive search plan which minimizes the probability of the HVU being killed by a hostile submarine.

Washburn (1983) showed that the FAB (Forward And Backward) algorithm could be applied to search for a target moving with a Markov process. Although he suggested that the FAB algorithm could be applied to a defensive optimal search, there has been no application of the FAB algorithm for this purpose.

## B. OBJECTIVE

This thesis demonstrates that the FAB algorithm is applicable to a kind of tactical decision aid which gives an optimal search plan to defend the HVU against a hostile submarine moving with a Markov process. A prototype of such a TDA using the FAB algorithm is described below.

# II. METHODOLOGY

## A. GENERAL SITUATION

The problem is to find an optimal search plan for minimizing the expected loss of the HVU against a hostile submarine, based on rough information on the submarine's location and direction of movement.

The assumptions are as follows:

- Time and space are discrete.
- A hostile submarine starts from an arbitrary point and moves with a Markov process.
- A friendly HVU tries to transit the submarine's patrol area.
- The HVU transits one unit of length per unit time.
- The submarine can use a USM (Undersea to Surface Missile) to kill the HVU within range.
- In addition, the submarine can use a USM and a torpedo to kill the HVU when within torpedo range (USM range is greater than torpedo range).
- The commander of the HVU has a limited number of ASW aircraft to search for and attack the enemy submarine.
- Each ASW aircraft is assigned a particular area(cell) and searches for the target randomly. If any aircraft detects the target, the target is considered killed.
- At the beginning, the commander has information on the target's location with a given error, and information on the target's motion based on the transition probabilities of a Markov process.

In this situation, the commander must decide how to search for the target.

Figure 1. shows these assumptions.

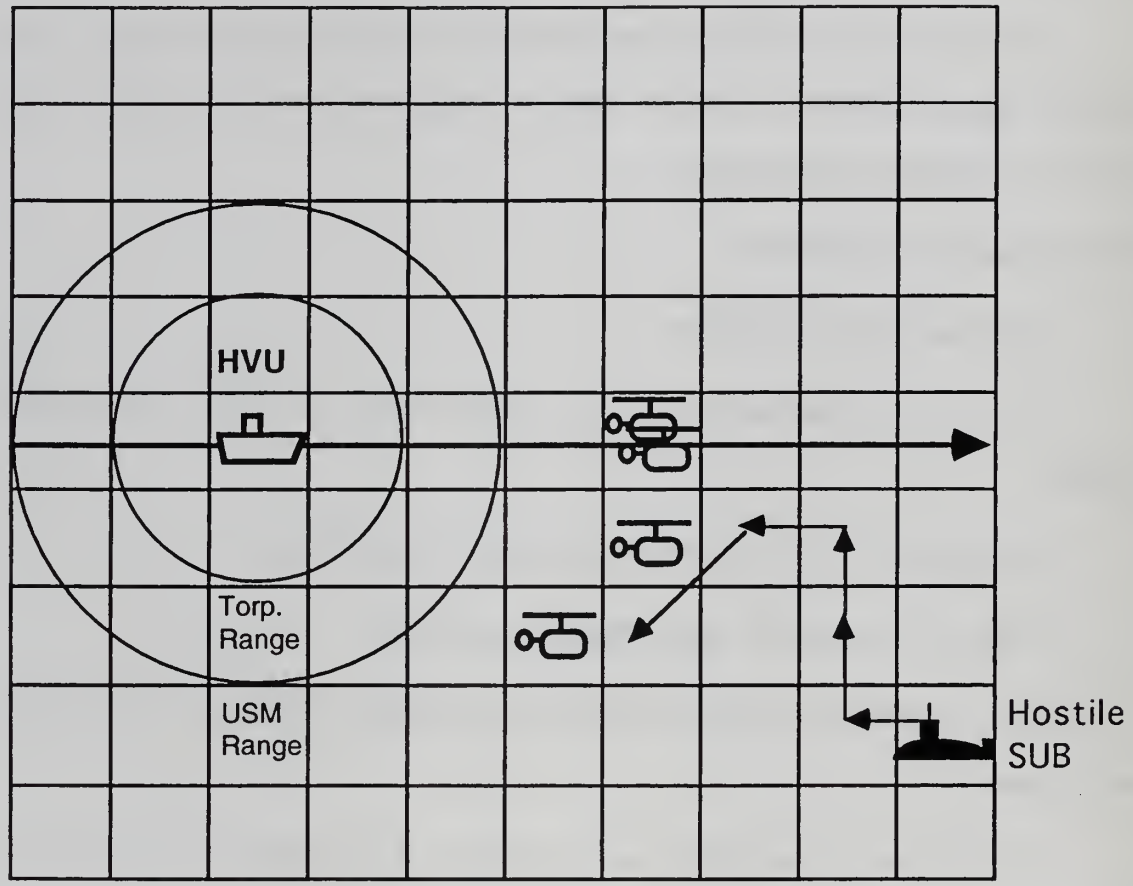


Figure 1. An illustration of the problem.

## B. PROBLEM FORMULATION

Let

$$X(t) = (x_t, y_t)$$

be a random variable which represents the position of the hostile submarine moving with a Markov process. The initial distribution and transition probabilities are known to be

$$\Pr\{X(t=0)=(x,y)\}=p(x,y)$$
$$\Pr\{X(t+1)=(x',y')\mid X(t)=(x,y)\}=\Gamma(x,y,x',y',t)$$

Let *n(X,t)* be the number of ASW aircraft assigned to each cell(X) at time t. These are the decision variables. Let *U(n,X,t)* be the probability that n search assets fail to detect the target in cell X at time t when the target is in the cell, and assume that

$$U(n(x,y,t),x,y,t)=\exp(-c(x,y,t)\cdot n(x,y,t)),$$

where *c(x,y,t)* is a search coefficient that reflects the acoustic conditions in the cell, the sensor performance, and the sweep rate of search units. To simplify, assume that *c(x,y,t)* is a constant *c* and omit the unnecessary arguments of *U( )*.

$$U(n(X,t))=\exp(-c\cdot n(X,t))$$

From the assumption in the previous section, the hostile submarine can use USMs and torpedoes to attack a surface unit. Let *D(X,t)* be the probability that the submarine at X is able to kill the HVU at time t, given that it decides to attack. Since $R_t < R_m$,

$$\begin{aligned} D(X,t) &= 1-(1-SSKP_t)\cdot(1-SSKP_m) && : \text{distance}(Sub.,HVU)\le R_t \\ &= SSKP_m && : R_t < \text{distance}(Sub.,HVU)\le R_m \\ &= 0 && : R_m < \text{distance}(Sub.,HVU) \end{aligned}$$

where $R_t$ and $R_m$ are ranges of the torpedo and the USM, $SSKP_t$ and $SSKP_m$ are the single shot kill probabilities of the torpedo and the missile.

Let *Y(t)* be the probability that the submarine will try to attack the HVU at time t

$$\sum_{t=0}^{t_{\max}} Y(t)=1.$$

Given that it survives, the submarine will attack the HVU exactly once. Let *L(X,t)* be the lethality defined as

$$L(x,y,t) = Y(t) \cdot D(x,y,t).$$

Since

$$\prod_{t'=0}^{t} U(n(X,t))$$

is the probability that the submarine survives to at least time t, the probability that the HVU is attacked is

$$H(n(\,)) = E\left[\sum_{t=0}^{t_{max}} L(X,t) \prod_{t'=0}^{t} U(n(X,t))\right].$$

The searcher's problem is therefore

$$\begin{aligned} &\underset{n(\,)}{Min}\ [H(n(\,))] \\ &\text{S/T}\ \sum_{x,y} n(x,y,t) \le N_{aircraft}\ \text{ for all } t \\ &n(x,y,t)\ \text{integers}\ \ge 0 \end{aligned}$$

## C. FAB ALGORITHM

Washburn [1] shows that *H(n( ))* can be represented as

$$H(n(\,)) = A(t) + \sum_{X} P(X,t) \cdot U(n(X,t)) \cdot F(X,t) \quad \text{for all t,} \tag{1}$$

where

$$A(t) = E\left[\sum_{t'=0}^{t-1} L(X,t) \cdot \prod_{t''=0}^{t'} U(n(X,t))\right].$$

In addition, the forward function *P( )* and the backward function *F( )* are defined as

follows:

$$\begin{cases} P(x,y,t=0) = p(x,y) & \text{for all } x,y \\ P(x,y,t+1) = \sum_{x',y'} \Gamma(x,y,x',y',t) \cdot U(n(x',y',t)) \cdot P(x',y',t) & \\ & \text{for all } x,y,0 \le t < t_{\max} \end{cases} \tag{2}$$

$$\begin{cases} F(x,y,t=t_{\max}) = L(x,y,t_{\max}) & \text{for all } x,y \\ F(x,y,t) = L(x,y,t) + \sum_{x',y'} \Gamma(x',y',x,y,t) \cdot U(n(x',y',t)) \cdot F(x',y',t+1) & \\ & \text{for all } x,y,0 < t \le t_{\max} \end{cases} \tag{3}$$

Now, assume that

$$n(t) = n'(t) \text{ for any } t \neq t_0. \tag{4}$$

Then,

$$H(n(\,)) - H(n'(\,)) = \sum_{x,y} P(x,y,t_0) \cdot \{U(n(x,y,t_0)) - U(n'(x,y,t_0))\} \cdot F(x,y,t_0). \tag{5}$$

(5) is true because from (1)-(4), *A(x,y,t)*, *P(x,y,t)* and *F(x,y,t)* don't depend on $n(t_0)$ and $n'(t_0)$. If *n( )* exists such that $H(n'(\,)) \geq H(n(\,))$ for all *n'( )* satisfying (4), this *n( )* is "critical." In addition, an allocation *n( )* is optimal if $H(n'(\,)) \geq H(n(\,))$ for all feasible *n'( )*. Criticality is necessary but not sufficient for optimality.

The main idea of the FAB algorithm is as follows:

If *n( )* is not critical, (4) and (5) guarantee that an allocation *n'( )* can be found which satisfies $H(n(\,)) > H(n'(\,))$. After solving the minimizing problem

$$\min_{n'(\,)} \sum_{X} P(X,t) U(n'(X,t)) F(X,t),$$

the FAB algorithm replaces n( ) with n'( ), calculates *P( )* and *F( )* based on *n'( )*, and repeats this process until *H( )* can't be reduced.

In pseudocode, The FAB algorithm is:

*function $update\_allocation(P(x,y,t), F(x,y,t))$
begin
  Min $\sum_{x,y} P(x,y,t) \cdot U(m(x,y,t)) \cdot F(x,y,t)$
  S/T $\sum_{x,y} m(x,y,t) \le N_{aircraft}$
  $m(x,y,t)$ integer $\ge 0$
 $update\_allocation \leftarrow m(\,)$
 return
*end function

*procedure FAB$(p(x,y), \Gamma(x',y',x,y,t),\ L(x,y,t))$
begin
 set $n(\,) = 0$ for all $x,y,t$
 calculate $F(x,y,t)$ with $n(\,)$ all $t$ using(3)
 $P(x,y,0)$ = InitialDistribution$(x,y)$
 Do
  $H_0 = H_1$
  calculate $F(x,y,t)$ with $n(\,)$ all $t$ using(3)
  for $t = 0$ to $t_{end}$
   $n(\,,t) \leftarrow update\_allocation(P(x,y,t), F(x,y,t))$
   calculate $P(x,y,t+1)$ with $n(\,,t)$ using(2)
  next $t$
  $H_1 = \sum_{x,y} P(x,y,0) \cdot U(n(x,y,0)) \cdot F(x,y,0)$
 Loop until $H_0 = H_1$
 output $n(\,)$, $H_0$
end procedure

Since the FAB algorithm starts by setting $n(\,)=0$, the first iteration produces an allocation that would be optimal if no searching were performed in the future. Such an allocation is called "myopic" and is of some interest in itself. In many cases, the myopic plan is nearly

optimal.

## D. DYNAMIC PROGRAMMING

In the FAB algorithm, it is necessary to solve the nonlinear integer minimizing problem in the function *update_allocation( )*. For this purpose, this research uses a dynamic programming procedure. By translating discrete x-y coordinates to the sequence 0 to (NxNy-1) like Figure 2., the problem can be restated as a 1-dimensional distribution problem.

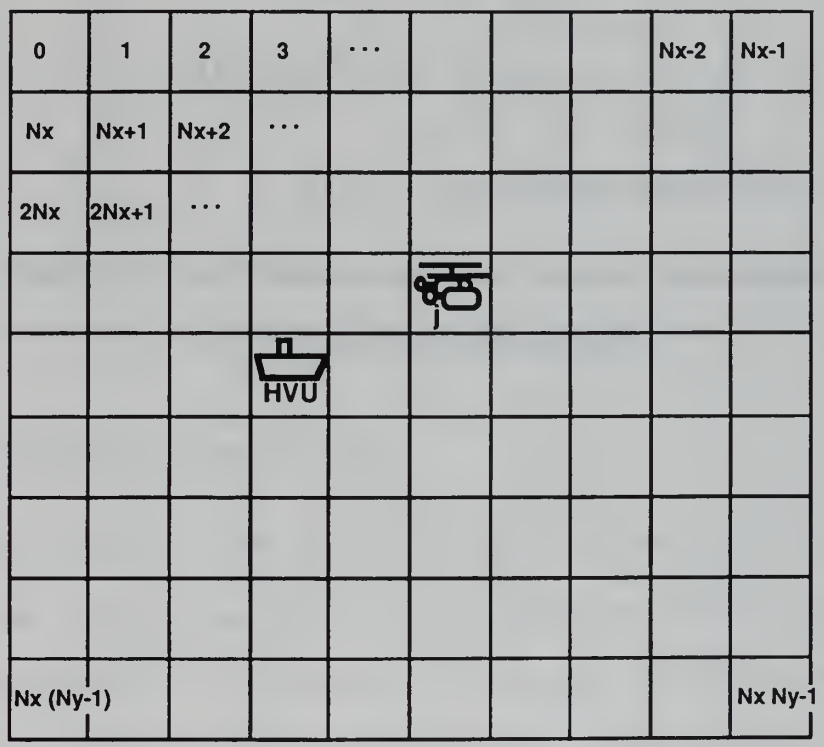


Figure 2. 1-dimensional indices for x-y coordinates.

$$\begin{pmatrix} x = i \bmod(N_x) \\ y = i \operatorname{div} N_x \end{pmatrix} \leftrightarrow i = y \cdot N_x + x$$

for all $t$

$$\begin{cases} \min\limits_{n(j,t)}\left( H(n) = \sum\limits_{j} P(j,t)U(n(j,t))F(j,t) \right) \\ S/T \\ \sum\limits_{j=0}^{N_x N_y - 1} n(j,t) \le N_{aircraft} \\ n(i,t) \text{ integers } \ge 0 \end{cases}$$

In recursive form, this problem is

$$\begin{cases} h_0(N) = \min\limits_{0 \le n \le N} \left[ P(0,t)U(n)F(0,t) \right] \\ \quad \text{for } j = 1 \cdots N_x N_y - 1 \\ \quad h_j(N) = \min\limits_{0 \le m+n \le N} \left[ h_{j-1}(m) + P(j,t)U(n)F(j,t) \right] \\ \min(H) = h_{N_x N_y - 1}(N_{aircraft}) \\ N, n, m \text{ integers } \ge 0 \end{cases}$$

Figure 3. illustrates how to solve this type of a minimizing problem using dynamic programming. (Sugiyama (1976)).

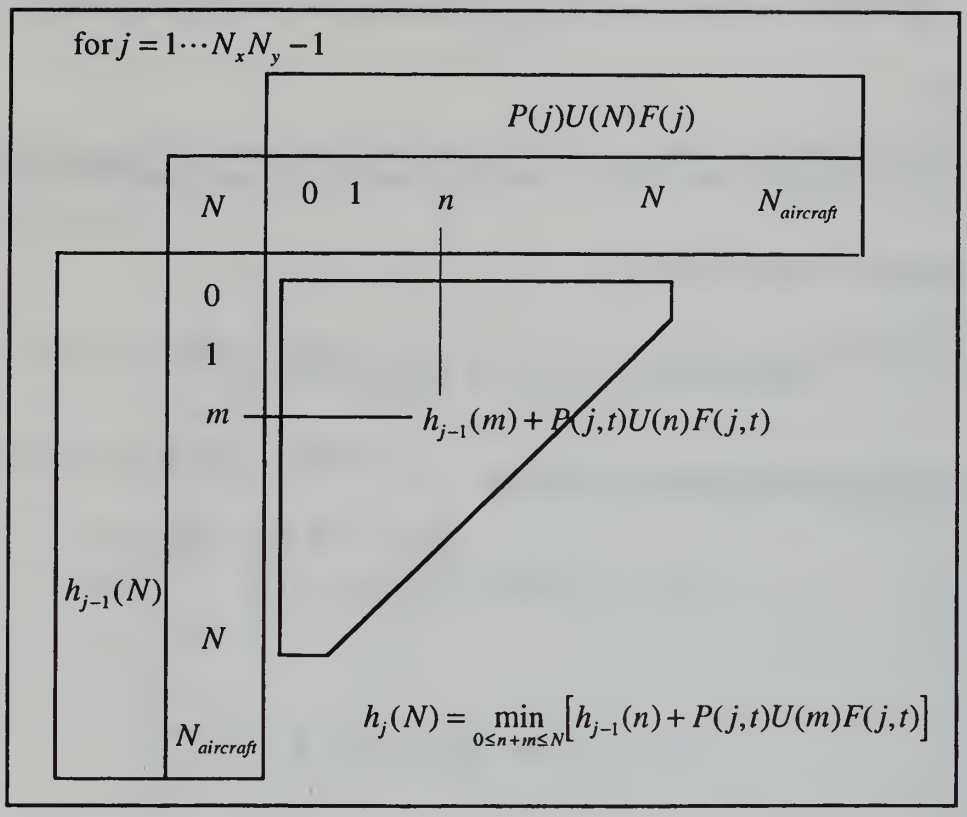


Figure 3. Minimizing the recursive form in dynamic programming. (Sugiyama (1976))

## E. A TWO SIDED GAME VALUE

A procedure to find the optimal allocation to protect the HVU from a hostile submarine has been described. However, this formulation includes a given *Y(t)* which represents the probability that the submarine tries to attack the HVU at time t. To clarify the dependence of *H* on *Y( )*, use explicit notation *H( ) = H(n( ),Y())* below. In the actual situation, the HVU commander can't know the submarine's intention, but the commander can assume that the submarine will maximize its chance to kill the HVU. Therefore, this situation is described as a two sided game. It follows that:

• The HVU commander controls $n(x,y,t)$ as decision variables to minimize $H(n(\ ),Y(\ ))$.

• The submarine controls $Y(t)$ as decision variables to maximize $H(n(\ ),Y(\ ))$.

The game value $v$ is in the interval

$$\underset{Y}{Max}\ \underset{n}{Min}\left[H(n(\ ),Y(\ ))\right] \le v \le \underset{n}{Min}\ \underset{Y}{Max}\left[H(n(\ ),Y(\ ))\right].$$

If $n_k(\ )$ is the allocation returned by FAB, let

$$f_t(n_k(\ )) \equiv E\left[D(X,t)\prod_{t'=0}^{t} U(n_k(X,t'))\right].$$

Since

$$H(n_k(\ ),Y(\ )) = \sum_{t=0}^{t_{max}} Y(t) f_t(n_k(\ ))\,,$$

$$\underset{Y}{Max}\left[H(n_k(\ ),Y(\ ))\right] \le \underset{t}{Max}\, f_t(n_k(\ ))\,.$$

Therefore, the value $Max\, f_t(n_k(\ ))$ is an upper bound on $v$.

To find the a lower bound on $v$, define $G(Y)$ as

$$G(Y) \equiv \underset{n}{Min}\left[H(n(\ ),Y(\ ))\right].$$

To simplify the exposition, assume for the moment that $G(Y)$ is the objective function returned by the FAB algorithm; that is, assume FAB returns a global minimum. Since $G(Y) \le v$ for every $Y$, any submarine strategy $Y$ has an associated lower bound $G(Y)$. To find good lower bound, $Y$ must be chosen to make $G(Y)$ large. Maximizing $G(Y)$ is difficult, but still a good value for $Y$ (call it $Y_k$) can be found using the following "greedy" algorithm where

$$\sum_{t=0}^{t_{\max}} Y(t)$$

is increased gradually from 0 to 1 with step $\Delta Y$. In each step, add $\Delta Y$ to $Y(t)$ which gives largest increment:

$$\underset{t}{Max}\left[\underset{n}{Min}\left[H(n(\ ),Y(0),Y(1),,,Y(t-1),Y(t)+\Delta Y,Y(t+1),,,Y(t\max))\right]\right].$$

In pseudocode, the greedy algorithm is:

```
*procedure  find_sub_strategy
begin
  Y_total = 0
  H_0 = 0
  Do while Y_total < 1
    for t = 0 to t_end
      Y_store ← Y(t)
      Y(t) ← Y(t) + ΔY
      call procedure FAB(p(x,y), Γ(x',y',x,y,t), Y(t)·D(x,y,t))
      if H > H_0 then
        H_0 ← H
        t_0 ← t
      end if
      Y(t) ← Y_store
    next t
    Y(t_0) ← Y(t_0) + ΔY
    Y_total ← Y_total + ΔY
  Loop
  output Y( ), H
end procedure
```

There is good reason to expect $Y_k$ to be nearly optimal. First, $G(Y)$ is a concave function because

$$G(\alpha \cdot Y_1 + (1-\alpha) \cdot Y_2) = \underset{n}{Min}\left[ \alpha \cdot \sum_{t=0}^{t_{max}} Y(t) \cdot f_t(n(X,t)) + (1-\alpha) \cdot \sum_{t=0}^{t_{max}} Y(t) \cdot f_t(n(X,t)) \right]$$

$$\geq \alpha \cdot \underset{n}{Min}\left[ \sum_{t=0}^{t_{max}} Y(t) \cdot f_t(n(X,t)) \right] + (1-\alpha) \cdot \underset{n}{Min}\left[ \sum_{t=0}^{t_{max}} Y(t) \cdot f_t(n(X,t)) \right]$$

$$= \alpha \cdot G(Y_1) + (1-\alpha) \cdot G(Y_2) \qquad \text{for } 0 < \alpha < 1$$

Second, if *G(Y)* were separable as

$$G(Y) = \sum_{t=0}^{t_{max}} G_t(Y(t)),$$

the greedy algorithm would find the $Max_Y$ *G(Y)*. Although *G(Y)* is not separable, at least *H(n( ),Y( ))* is separable, so there is good reason to expect $Y_k$ to be nearly optimal. A lower bound is obtained in any case. Some numerical examples will be given in chapter ⌋ . In practice the lower bound may be in error to the extent that the value returned by the FAB algorithm is not a true minimum.

## F. IMPLEMENTATION

The prototype FABTDA is a Microsoft Windows application written in Visual Basic ver. 3.0. FABTDA has the following functions:

• FABTDA finds the optimal allocation plan of ASW assets for a given *Y(t)* with the FAB algorithm.

• FABTDA finds a myopic allocation plan of ASW assets for a given *Y(t)*.

• FABTDA finds a submarine strategy which gives the lower bound on the game value.

• FABTDA finds an upper bound on the game value.

• FABTDA finds an optimal offensive search plan, where the word "offensive" means that the search minimizes the probability that the submarine survives until the last

period without regard to the fate of the HVU. The offensive search plan can be calculated based on $D(x,y,t)=1$ and $Y(t)=1$ if $t=t_{max}$, $Y(t)=0$ if $t<t_{max}$. FABTDA also shows the corresponding probability that the HVU is attacked based on the offensive search plan.

Figure 4 illustrates the relationship of the procedures, inputs and outputs of FABTDA.

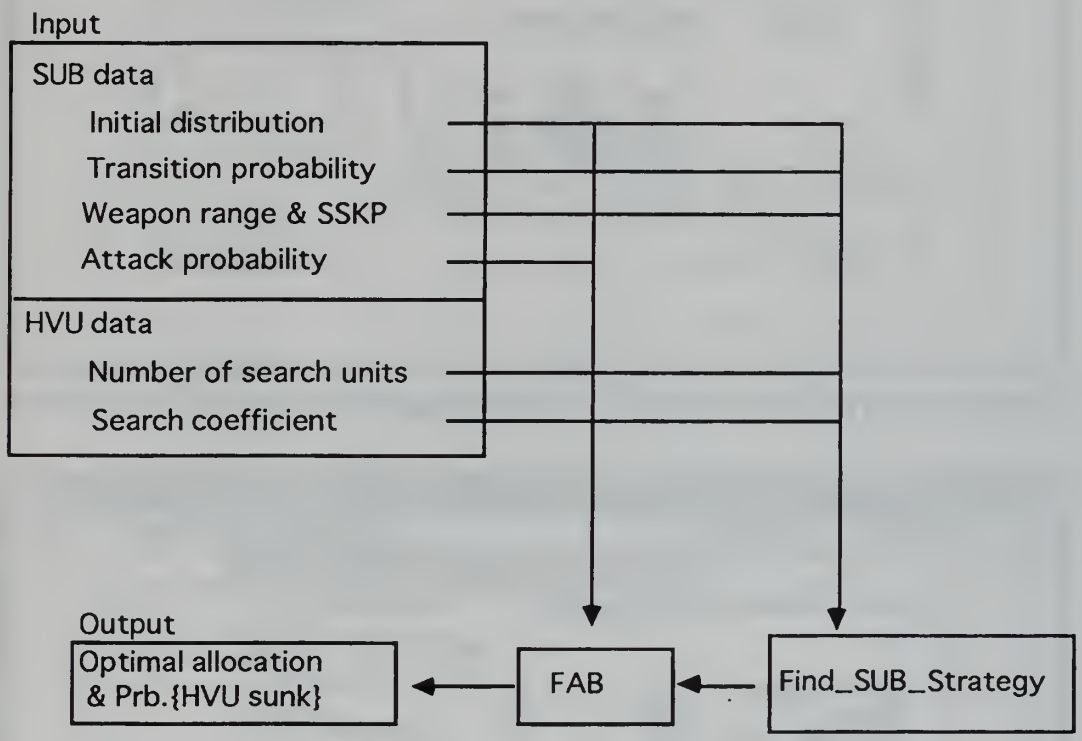


Figure 4 Outline of the prototype TDA: FABTDA.

Figures 5 through 9 show data input and output forms of FABTDA. The data can be changed by mouse on these input forms. All data can be stored to or loaded from a file by selecting the menu "file." If the user clicks a command button, corresponding values are sent from the input form to the variable in the program. Then the caption of the command button "data set" is changed into "set done."

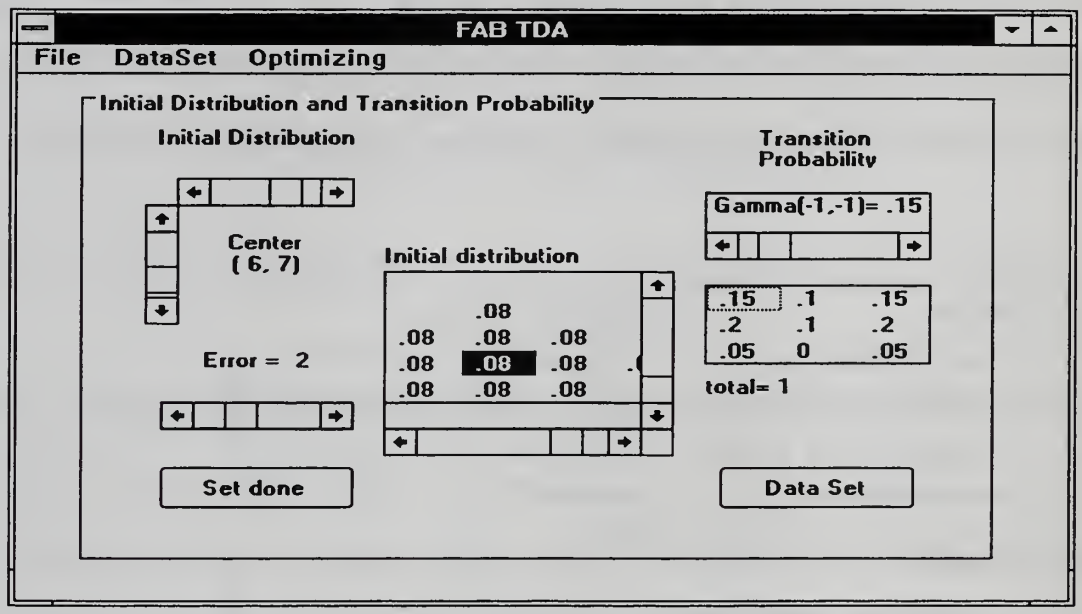


Figure 5. FABTDA input form for initial distribution and transition probability.

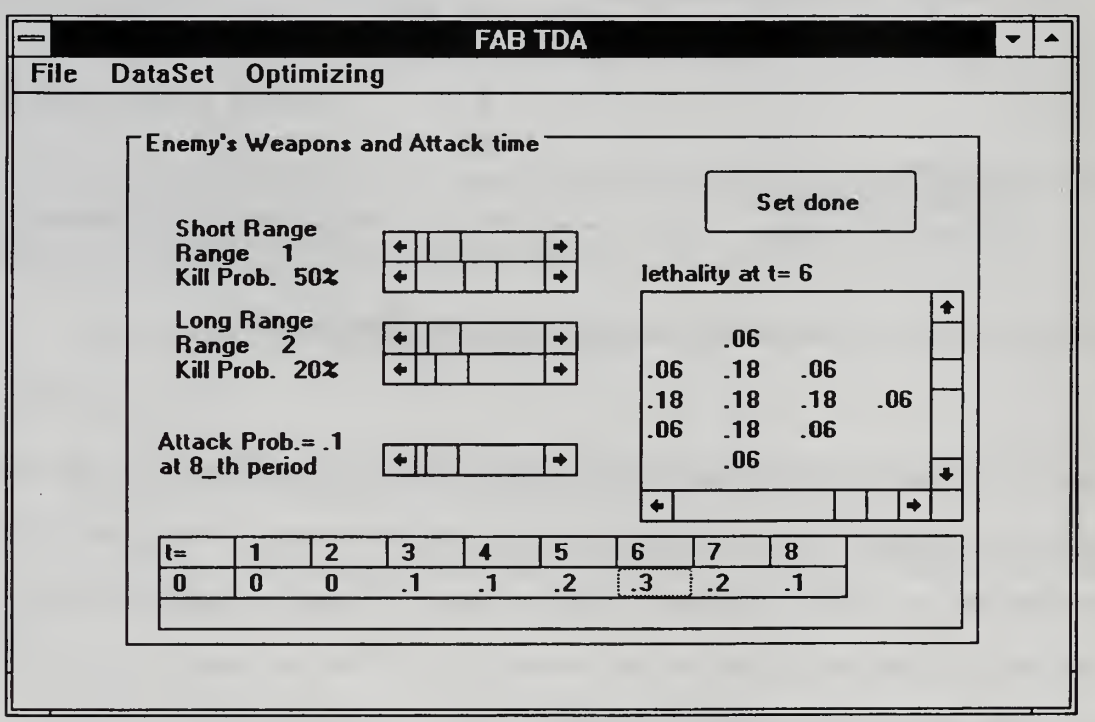


Figure 6. FABTDA input form for weapon attributes and attack probabilities.

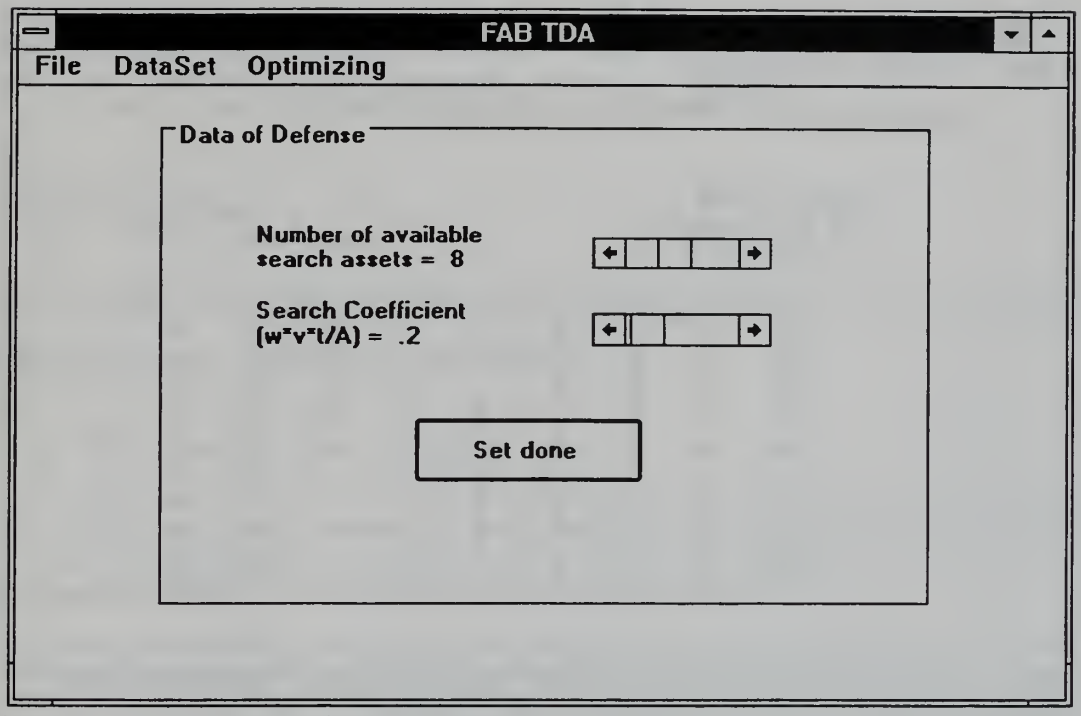


Figure 7. FABTDA input form for number of ASW assets and their search capability.

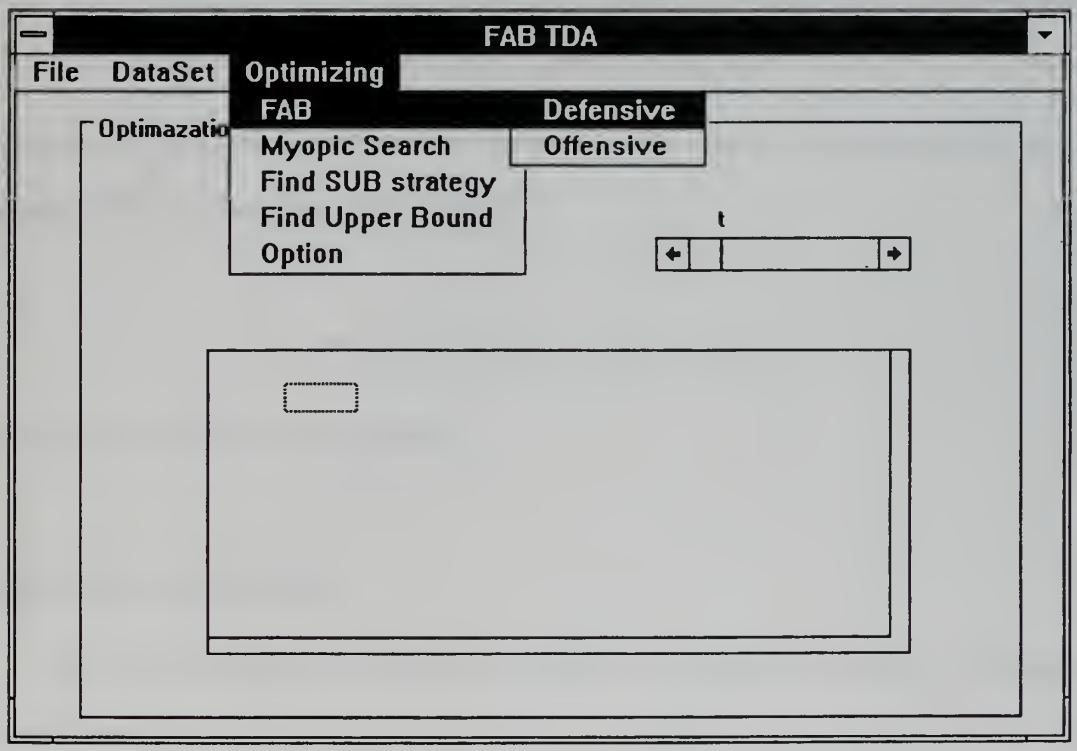


Figure 8. FABTDA with the defensive option being selected.

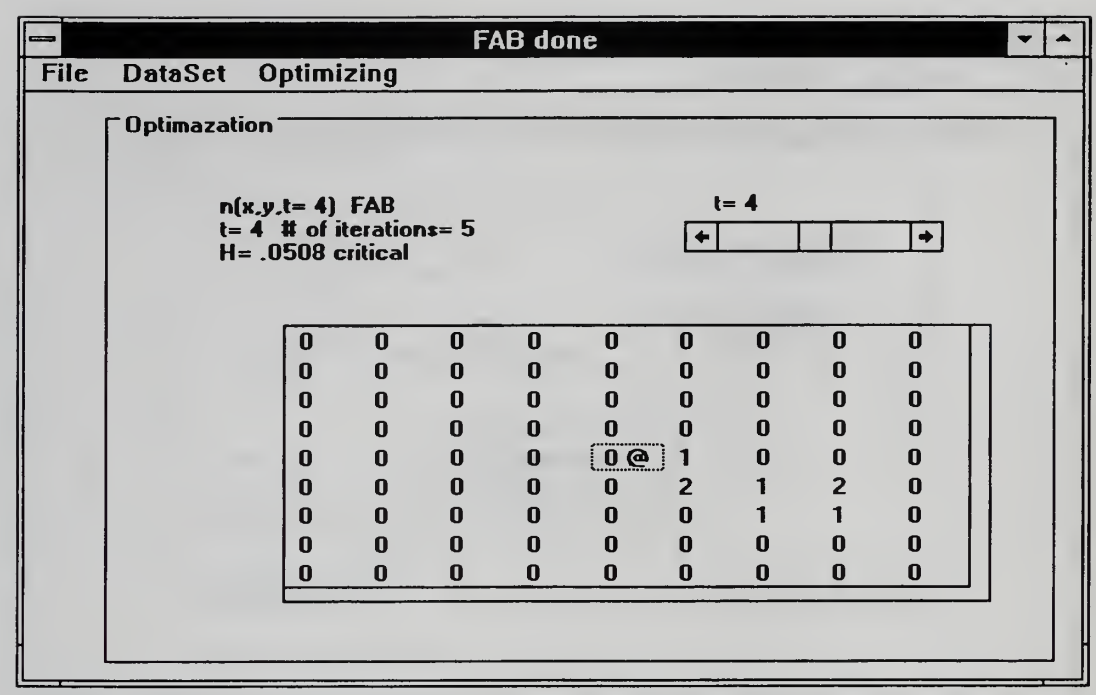


Figure 9. FABTDA output form for an optimal allocation by the FAB algorithm.

The allocation of search units at time t is indicated on the grid. Moving the scroll bar changes the value of t. The "@" symbol shows the position of the HVU.

# III. NUMERICAL EXAMPLES

## A. VERIFICATION

To verify FABTDA, consider a simple example which can be solved analytically. For this purpose, use the following assumptions:

- Time periods are zero to eight.
- The submarine is a stationary target, and may be in one of 5 cells: datum center and its nearest 4 cells.
- The submarine's weapon range is infinity, and its SSKP=1.
- The submarine's attack intention Y(t) =1 if t=8, Y(t) =0 if t≠8.
- The HVU has 5 ASW assets, and each has a 0.2 search coefficient.

Obviously, the optimal allocation is to assign one ASW unit in each of 5 cells, and the minimum objective function value should be

$$\min_{n}(H) = 5 \times \frac{1}{5} \times e^{-(0.2*9)} \approx 0.1653 .$$

The output of FABTDA is as expected.

## B. ACTUAL EXAMPLE

The next example deals with a more realistic situation. The realistic assumptions are as follows:

- Space is discrete with a 9 by 9 grid, and the cell size is 15nm x 15nm.

• Time is also discrete running from 0 to 9 in one hour periods.

• The HVU transits at 15kt, moving a single cell in each period.

• The hostile submarine patrols the area with following transition probability,

$$\Pr\{X(t+1)=(x+\Delta_x, y+\Delta_y) \mid X(t)=(x,y)\} = \begin{array}{c} \Delta_x=-1, \quad 0, \quad 1 \\ \begin{pmatrix} .05 & .15 & .3 \\ .05 & .1 & .25 \\ 0 & .05 & .05 \end{pmatrix} \end{array} \begin{array}{c} \\ \Delta_y=-1 \\ 0 \\ 1 \end{array}$$

On the boundary, the submarine turns back to the inside.

• Initially, the HVU has information that the submarine is 45nm ahead and 45nm starboard with a 30nm error (see initial distribution below).

• The submarine has torpedoes with a 15nm range and a 70% SSKP and also has USMs with a 45nm range and a 20% SSKP.

• The probabilities of submarine attack during each period are 0, 0, 0.1, 0.1, 0.2, 0.2, 0.2, 0.1, and 0.1.

• The HVU has 8 ASW helicopters. Each one can effectively sweep 20% of the 15nm x 15nm area in one hour.

Using these settings, FABTDA gives the following output:

```
This is an outcome of FAB_TDA.
 by M.KUNIGAMI

General settings

   matrix size
     (0 to 8) * (0 to 8)
   time
      0 to 8

Initial enemy data.
```

```
initial distribution
          x= 0  x= 1  x= 2  x= 3  x= 4  x= 5  x= 6  x= 7  x= 8
 y=  0    .0    .0    .0    .0    .0    .0    .0    .0    .0
 y=  1    .0    .0    .0    .0    .0    .0    .0    .0    .0
 y=  2    .0    .0    .0    .0    .0    .0    .0    .0    .0
 y=  3    .0    .0    .0    .0    .0    .0    .0    .0    .0
 y=  4    .0    .0    .0    .0    .0    .0    .0    .0    .0
 y=  5    .0    .0    .0   .083   .0    .0    .0    .0    .0
 y=  6    .0    .0   .083  .083  .083   .0    .0    .0    .0
 y=  7    .0   .083  .083  .083  .083  .083   .0    .0    .0
 y=  8    .0    .0   .083  .083  .083   .0    .0    .0    .0

transition probability to neighbors
          x=-1  x= 0  x= 1
 y=-1     .05   .15   .3
 y= 0     .05   .1   .25
 y= 1     .0    .05   .05

weapon attributions
   short range weapon:
         range = 1,  SSKP= .7
   long range weapon :
         range = 3,  SSKP= .2

 prob. that SUB trys to attack
          t= 0  t= 1  t= 2  t= 3  t= 4  t= 5  t= 6  t= 7  t= 8
      p=    .0    .0    .1    .1    .2    .2    .2    .1    .1

Search units data.
   Number of available search units : 8
   sweep rate of each unit : .2


Outcome: allocation plan of search units by FAB Defensive
        @s indicate position of HVU
Objective function value :H= .10375

 t= 0
          x= 0 x= 1 x= 2 x= 3 x= 4 x= 5 x= 6 x= 7 x= 8
 y=  0    0    0    0    0    0    0    0    0    0
 y=  1    0    0    0    0    0    0    0    0    0
 y=  2    0    0    0    0    0    0    0    0    0
 y=  3    0    0    0    0    0    0    0    0    0
 y=  4    0 @  0    0    0    0    0    0    0    0
 y=  5    0    0    0    2    0    0    0    0    0
 y=  6    0    0    2    2    1    0    0    0    0
 y=  7    0    1    0    0    0    0    0    0    0
 y=  8    0    0    0    0    0    0    0    0    0

 t= 1
          x= 0 x= 1 x= 2 x= 3 x= 4 x= 5 x= 6 x= 7 x= 8
 y=  0    0    0    0    0    0    0    0    0    0
 y=  1    0    0    0    0    0    0    0    0    0
 y=  2    0    0    0    0    0    0    0    0    0
 y=  3    0    0    0    0    0    0    0    0    0
 y=  4    0    0 @  0    0    0    0    0    0    0
```

```
y=  5   0    0    0    0    0    0    0    0    0
y=  6   0    0    1    2    1    0    0    0    0
y=  7   0    0    0    2    2    0    0    0    0
y=  8   0    0    0    0    0    0    0    0    0

t= 2
       x= 0 x= 1 x= 2 x= 3 x= 4 x= 5 x= 6 x= 7 x= 8
y=  0   0    0    0    0    0    0    0    0    0
y=  1   0    0    0    0    0    0    0    0    0
y=  2   0    0    0    0    0    0    0    0    0
y=  3   0    0    0    0    0    0    0    0    0
y=  4   0    0    0 @  0    0    0    0    0    0
y=  5   0    0    0    1    2    0    0    0    0
y=  6   0    0    0    2    2    1    0    0    0
y=  7   0    0    0    0    0    0    0    0    0
y=  8   0    0    0    0    0    0    0    0    0

t= 3
       x= 0 x= 1 x= 2 x= 3 x= 4 x= 5 x= 6 x= 7 x= 8
y=  0   0    0    0    0    0    0    0    0    0
y=  1   0    0    0    0    0    0    0    0    0
y=  2   0    0    0    0    0    0    0    0    0
y=  3   0    0    0    0    0    0    0    0    0
y=  4   0    0    0    0 @  1    0    0    0    0
y=  5   0    0    0    1    2    1    0    0    0
y=  6   0    0    0    0    2    1    0    0    0
y=  7   0    0    0    0    0    0    0    0    0
y=  8   0    0    0    0    0    0    0    0    0

t= 4
       x= 0 x= 1 x= 2 x= 3 x= 4 x= 5 x= 6 x= 7 x= 8
y=  0   0    0    0    0    0    0    0    0    0
y=  1   0    0    0    0    0    0    0    0    0
y=  2   0    0    0    0    0    0    0    0    0
y=  3   0    0    0    0    0    0    0    0    0
y=  4   0    0    0    0    1 @  2    0    0    0
y=  5   0    0    0    0    3    1    1    0    0
y=  6   0    0    0    0    0    0    0    0    0
y=  7   0    0    0    0    0    0    0    0    0
y=  8   0    0    0    0    0    0    0    0    0

t= 5
       x= 0 x= 1 x= 2 x= 3 x= 4 x= 5 x= 6 x= 7 x= 8
y=  0   0    0    0    0    0    0    0    0    0
y=  1   0    0    0    0    0    0    0    0    0
y=  2   0    0    0    0    0    0    0    0    0
y=  3   0    0    0    0    0    0    0    0    0
y=  4   0    0    0    0    0    1 @  2    1    0
y=  5   0    0    0    0    0    3    0    1    0
y=  6   0    0    0    0    0    0    0    0    0
y=  7   0    0    0    0    0    0    0    0    0
y=  8   0    0    0    0    0    0    0    0    0

t= 6
       x= 0 x= 1 x= 2 x= 3 x= 4 x= 5 x= 6 x= 7 x= 8
y=  0   0    0    0    0    0    0    0    0    0
y=  1   0    0    0    0    0    0    0    0    0
```

```
y=  2   0    0    0    0    0    0    0    0    0
y=  3   0    0    0    0    0    0    0    0    0
y=  4   0    0    0    0    0    0    0 @  4    0
y=  5   0    0    0    0    0    0    2    2    0
y=  6   0    0    0    0    0    0    0    0    0
y=  7   0    0    0    0    0    0    0    0    0
y=  8   0    0    0    0    0    0    0    0    0

t= 7
        x= 0 x= 1 x= 2 x= 3 x= 4 x= 5 x= 6 x= 7 x= 8
y=  0   0    0    0    0    0    0    0    0    0
y=  1   0    0    0    0    0    0    0    0    0
y=  2   0    0    0    0    0    0    0    0    0
y=  3   0    0    0    0    0    0    0    2    0
y=  4   0    0    0    0    0    0    0    3 @  0
y=  5   0    0    0    0    0    0    0    3    0
y=  6   0    0    0    0    0    0    0    0    0
y=  7   0    0    0    0    0    0    0    0    0
y=  8   0    0    0    0    0    0    0    0    0

t= 8
        x= 0 x= 1 x= 2 x= 3 x= 4 x= 5 x= 6 x= 7 x= 8
y=  0   0    0    0    0    0    0    0    0    0
y=  1   0    0    0    0    0    0    0    0    0
y=  2   0    0    0    0    0    0    0    0    0
y=  3   0    0    0    0    0    0    0    0    1
y=  4   0    0    0    0    0    0    0    4    1 @
y=  5   0    0    0    0    0    0    0    0    2
y=  6   0    0    0    0    0    0    0    0    0
y=  7   0    0    0    0    0    0    0    0    0
y=  8   0    0    0    0    0    0    0    0    0
```

In this example, the minimized defensive probability that the HVU is attacked is 0.1038. The same probability is 0.1145 if the HVU uses the offensive allocation; the increase shows the importance of choosing the right objective.

For the myopic defensive search, the probability is 0.1052, and for the myopic offensive search, the probability is 0.1159.

| Min H | defensive | offensive |
|---|---|---|
| FAB | 0.1038 | 0.1145 |
| Myopic | 0.1052 | 0.1159 |

The FAB algorithm finds a more effective search allocation than that from the myopic search.

In addition, FABTDA can provide a submarine strategy by the procedure *find_sub_strategy*. The procedure *find_sub_strategy* with $\Delta Y=0.1$ gives the following probability that the submarine will attack the HVU at t:

```
prob. that SUB tries to attack
      t=  0  t= 1  t= 2  t= 3  t= 4  t= 5  t= 6  t= 7  t= 8
      p= .0    .0    .0    .0    .1    .3    .3    .3    .0
```

Using this strategy, FAB fixes the HVU strategy to minimize the defensive objective function value. In this case, the objective function value is $H = 0.1133$, an improvement from the submarine's viewpoint. This value will be a lower bound on the game value as long as FAB obtains a nearly optimal solution.

Additionally, FABTDA provides an upper bound for the game value. In accordance with the submarine strategy, the upper bound of the game value is

$$f_7(n(\,)) = 0.1145.$$

Similar calculation can be performed for the offensive objective function.

In summary, the bounds are:

| Bounds of game value | FAB defensive | FAB offensive |
|---|---|---|
| Lower bound | 0.1133 | 0.1238 |
| Upper bound | 0.1145 | 0.1272. |

The upper bound of the defensive search is smaller than the lower bound of the offensive search. FABTDA provides good bounds that can discriminate between defensive search and offensive search. In this case, the defensive search is worth considering to defend the HVU. The best defense is not always a good offense.

# IV. CONCLUSIONS

FABTDA demonstrates that it is feasible to develop a TDA to find the optimal search plan defending the HVU for a moving target by using the FAB algorithm. In addition, the previous chapter shows that:

- The FAB algorithm can find more effective search plans than the myopic search.
- FABTDA can provide bounds on the game value, which show when the defensive search is more effective than the offensive search.

Currently, FABTDA uses a search coefficient that is a constant. In the future, if FABTDA can use space and time dependent search coefficients, more interesting allocation plans which reflect actual situations like acoustic conditions could be generated.

## LIST OF REFERENCES

Sugiyama, S. "Dynamic Programming," Nikkagiren, Tokyo, 1976.

Washburn, A. R., "Search for Moving target: The FAB Algorithm," Operations Research, Vol. 31, No. 4, pp. 739-751, 1983.

Washburn, A. R., "Search and Detection 2nd ed.," ORSA, 1989.

## INITIAL DISTRIBUTION LIST

No. Copies

1. Defense Technical Information Center ..... 2
   8725 John J. Kingman Rd., STE 0944
   Fort Belvoir, VA 22060-6218

2. Dudley Knox Library ..... 2
   Naval Postgraduate School
   411 Dyer Rd.
   Monterey, CA 93943-5101

3. Prof. A.R. Washburn, Code OR/Ws ..... 2
   Naval Postgraduate School
   Monterey, CA 93943-5002